

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application No. : 10/672,697 Confirmation No. 9222
Applicant : Vincent J. Zimmer
Filed : 9/26/2003
TC/A.U. : 2114
Examiner : Gabriel L. Chu

Docket No. : 042390.P16549
Customer No. : 8791

Commissioner for Patents
PO Box 1450
Alexandria VA 22313-1450

DECLARATION UNDER 37 C.F.R. § 1.131

Dear Sir:

I, Vincent J. Zimmer hereby declare that:

1. I am the inventor of the subject matter claimed in the above-identified patent application, which is assigned to Intel Corporation.
2. This declaration is to establish conception of the invention in the above-identified patent application in the United States, at a date prior to September 16, 2003, the issue date of U.S. Patent No. 6,622,260, which was cited by the Examiner.
3. I understand that the invention relates to the following:

A. A method comprising:
responsive to a platform error at a local node of a platform, performing error recovery at a processor abstraction layer (PAL);
if the platform error is not resolved at the PAL,
determining if there is a peer node with an available network interface card (NIC),
and if there is a peer node with an available NIC,
sending a media access control (MAC) address of the local node to the peer node so that the peer node can handle operations for the local node, and
disabling the MAC address of the local node, and
performing error recovery at a system abstraction layer (SAL);
if the platform error is resolved by the SAL,
enabling the local node with the MAC address of the local node,
the local node to resume normal operation.

B. A machine-readable medium having stored thereon instructions, which when executed by a machine, cause the machine to perform the following operations comprising:
responsive to a platform error at a local node of a platform, performing error recovery at a processor abstraction layer (PAL);
if the platform error is not resolved at the PAL,
determining if there is a peer node with an available network interface card (NIC),
and if there is a peer node with an available NIC,
sending a media access control (MAC) address of the local node to the peer node so that the peer node can handle operations for the local node, and
disabling the MAC address of the local node, and
performing error recovery at a system abstraction layer (SAL);
if the platform error is resolved by the SAL,
enabling the local node with the MAC address of the local node, the local node to resume normal operation.

C. A server blade comprising:
a processor;
a memory coupled to the processor; and
a network interface card (NIC) coupled to the processor to provide for network communications to a peer server blade;
wherein responsive to a platform error at the server blade, error recovery is performed at a processor abstraction layer (PAL) and if the platform error is not resolved at the PAL, a media access control (MAC) address of the server blade is sent to the peer server blade so that the peer server blade can handle operations for the server blade, and the MAC address of the server blade is disabled.

D. A server platform comprising:
a server blade rack;
a local server blade coupled to the server blade rack, the local server blade operating in conjunction with firmware; and
a peer server blade coupled to the server blade rack, the peer server blade operating in conjunction with firmware;
wherein responsive to a platform error at the local server blade, error recovery is performed at a processor abstraction layer (PAL) and if the platform error is not resolved at the PAL, a media access control (MAC) address of the local server blade is sent to the peer server blade so that the peer server blade can handle operations for the local server blade and the MAC address of the local server blade is disabled.

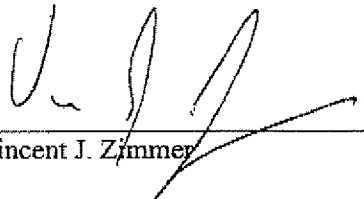
4. Prior to September 16, 2003, I completed an Invention Disclosure (Exhibit A) describing the invention and submitted the invention disclosure to the legal department of Intel Corporation.

5. After receipt and review of the Invention Disclosure, the legal department of Intel Corporation decided to proceed with the preparation of a patent application and requested that Blakely, Sokoloff, Taylor & Zafman LLP prepare and file a patent application on the subject matter set forth in Exhibit A.
6. Thereafter, the above-identified patent application was prepared with due diligence and filed on August 26, 2003.

I hereby declare that all statements made herein of my own knowledge are true and that the statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Date: _____

11/30/07



Vincent J. Zimmer

EXHIBIT A

INTEL INVENTION DISCLOSURE
ATTORNEY-CLIENT PRIVILEGED COMMUNICATION
 located at <http://legal.intel.com/patent/index.htm>

30147

DATE: 2/15/2003

SOFTWARE/EPG/PPG/EPD

It is important to provide accurate and detailed information on this form. The information will be used to evaluate your invention for possible filing as a patent application. **Invention Disclosure forms MUST be sent electronically via email to your manager/supervisor who should then forward with their approval to our email account "invention disclosure submission."** If you have any questions, please call :

Last Name: Komarla	First Name: Eshwari	M.I. P.
--------------------	---------------------	---------

Last Name: Zimmer	First Name: Vincent	M.I. J
-------------------	---------------------	--------

(PROVIDE SAME INFORMATION AS ABOVE FOR EACH ADDITIONAL INVENTOR)

2. Title of Invention:

~~MeWin to protect/separate phases of code in the code based on the Intel code (as specific as you can)~~

4. Include several key words to describe the technology area of the invention in addition to # 3 above:
 Firmware. Forms. BIOS. Device Driver. pre-boot. seal-healing. autonomic systems

5. Stage of development (i.e. % complete, simulations done, test chips if any, etc.):
Initial Design, prototyping

6a. Has a description of your invention been (or planned to be) published outside of Intel:
 No

If YES, was the manuscript submitted for pre-publication approval through the Author Incentive Program:

If YES, please identify the publication and the date published:

6b. Has your invention been used/sold or planned to be used/sold by Intel or others?

If YES, date it was sold or will be sold:

6c. Does this invention relate to technology that is or will be covered by a SIG (special interest group)/standard or specification?
 No

If YES, name of SIG/standard/specification:

6d. If the invention is embodied in a semiconductor device, actual or anticipated date of tapeout?
No

6e. If the invention is software, actual or anticipated date of any beta tests outside Intel:

7. Was the invention conceived or constructed in collaboration with anyone other than an Intel blue badge employee or in performance of a project involving entities other than Intel (e.g. government, other companies, universities or consortia)? NO: X If YES, name of individual or entity:

8. Is this invention related to any other invention disclosure that you have recently submitted? If so, please give the title and inventors: No

**PLEASE READ AND FOLLOW THE DIRECTIONS ON
HOW TO WRITE A DESCRIPTION OF YOUR INVENTION**

**Try to limit your description to 2-3 pages
Do NOT attach a presentation, white paper, or specification
ANSWER ALL OF THE QUESTIONS BELOW**

Please provide a description of the invention and include the following information:

1. Describe in detail what the components of the invention are and how the invention works.

This disclosure describes a method to have seamless failover of Itanium-based blades in a modular computing environment. This fail-over policy is effected through a combination of an Out-Of-Band (OOB) channel and exchanging networking interface controller (NIC) addresses between Itanium-based blades. This particular art is focused upon the Itanium blades because of the stylized error-recovery architecture. The type of system being described can be found in Figure1.

Specifically, there are architecturally-defined flows in an Itanium firmware stack where the platform code takes control upon receipt of an error, such as Machine Check Abort (MCA); then, there is a point where the firmware hand-shakes with the operating system in order to let the OS attempt error recovery. In the case of the former control point, the firmware can "blank" or disable its network and convey its network identity, via the Media Access Control (MAC) address to a peer node; the former can "unblank" during the latter control point when the operating system retrieves the error information. This allows for another unit to take over the network ID and traffic of a unit that is engaged in error-containment (i.e., time interval between blank and unblank, which can range from seconds to several ten's of minutes).

The problem being addressed herein is that during a platform failure, the firmware is signaled immediately via the MCA, but if the OS is unable to recover from the error, system takes a fatal error (bug check) followed by dumping of core. During the subsequent boot sequence, OS gathers error information from the firmware. Depending on the configuration of system and the applications running, the latency between occurrences of a fatal error to the time system is fully operational could range from seconds to several minutes. This latency can cause host requests to be lost or queued up, etc. This art will allow for firmware to seamlessly pass its network identity to a peer. The event flow including "blanking" and "unblanking" is shown in the Figure 2. For stateless protocols like HTTP and a rack-configuration of front-end web servers, this art should suffice to provide continual responsiveness. In addition, for load-balancing schemes like Round-Robin DNS (RR-DNS) with a Cisco Local Directory, for example, there would be little to no perturbation of system behavior.

This art is non-obvious in that it is using two architectural hook-points in the Itanium firmware flow and the mutable/shareable nature of network identities to provide this system-wide autonomic/self-healing enterprise system behavior.

2. Describe advantage(s) of your invention over what is currently being done.

In Summary:

- Today, if a system fails over, the Itanium architecture allows for several elaborate error-containment stages. These include PAL, SAL, and OS. The problem is that the SAL/Platform logic and the OS logic can be long-lived (i.e., perform PCI bus walk, interrogate 10's of devices, etc). This art will take the unit offline in firmware but pass its network ID to peer so that the latency of error containment is shrouded.

- This art could be used on IA32 with SMI-based error trapping and period SMI's to resume the control, but most advantage would be accrued for the Itanium units.

3. You MUST include at least one figure illustrating the invention. If the invention relates to

software, include a flowchart or pseudo-code representation of the algorithm.

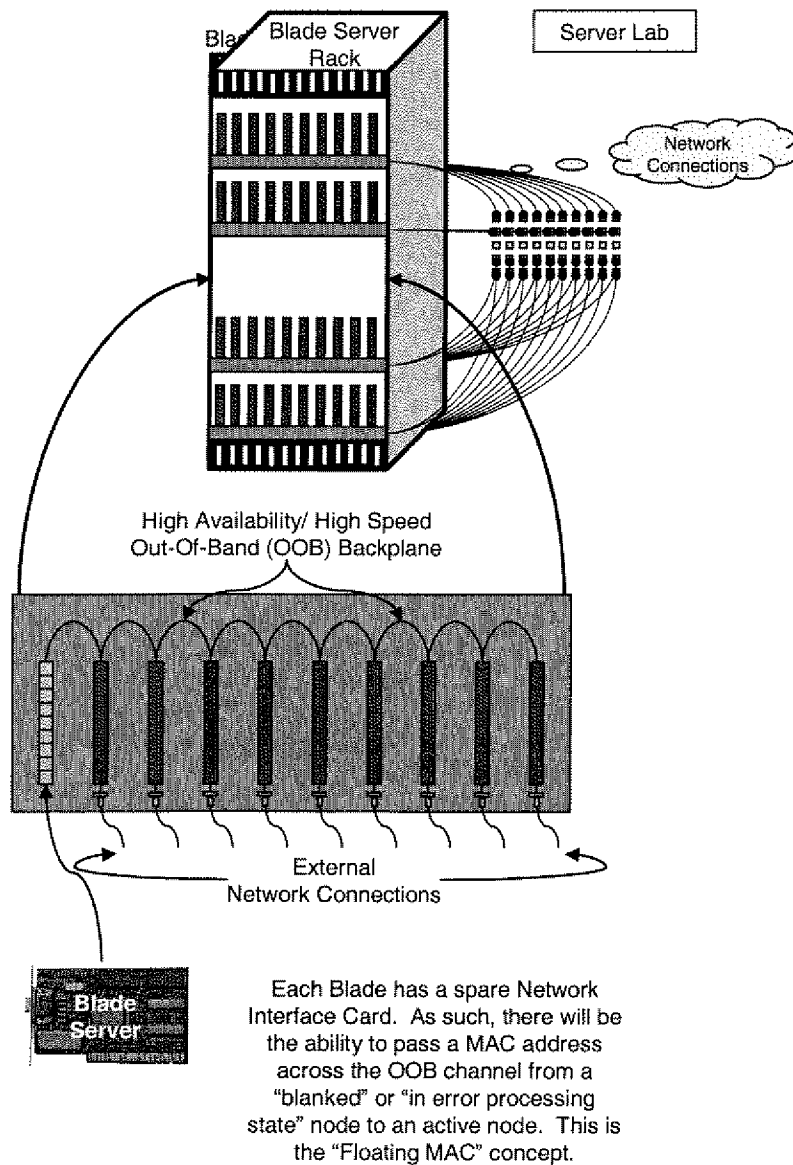


Figure 1 Blade Topology

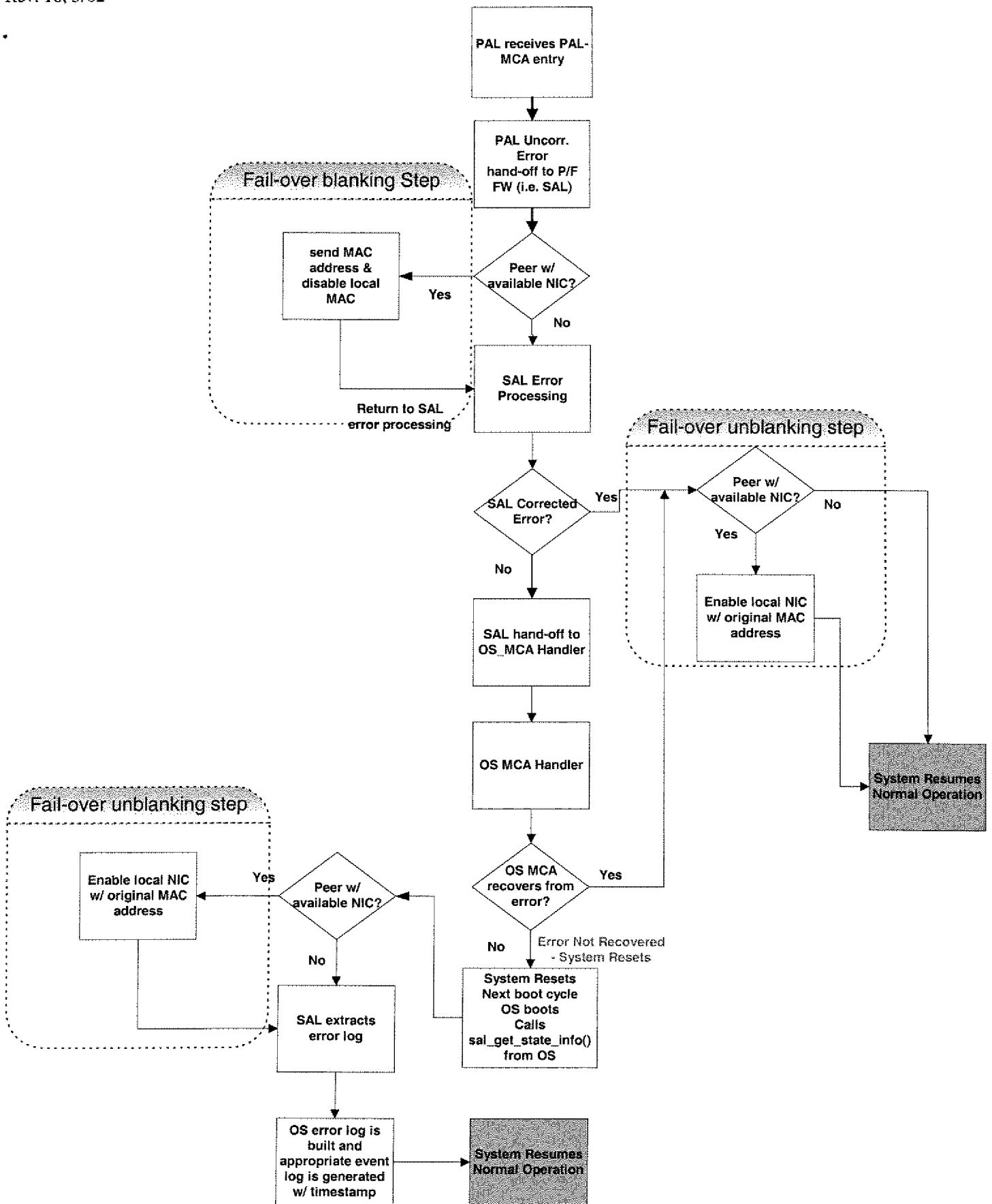


Figure 3 Flow Chart of Fail-Over Algorithm

4. Value of your invention to Intel (how will it be used?).

In Summary:

- This invention will allow Intel to provide a feature in its blade-based computing that other blade vendors cannot. The time-lag associated with long-lived actions like error-containment, etc., can be masked by having a peer computational unit take control. This will allow Intel as a supporter and originator of standards in this space to have more prevalence.

- As Intel builds Itanium systems, Itanium processors, NIC's, and authors standards for modular computing, this will help our customer meet five-9's of uptime and have the rack act as a single, consistent whole.

5. Explain how your invention is novel. If the technology itself is not new, explain what makes it different.

In Summary:

- Employ a "floating MAC" scheme to make peer unit's NIC look like NIC of unit in error-handling. The rack of systems looks more like an organic whole.

- Allow for constant, always-on network availability of the nodes. For front-end web-servers with many peer, identical front-end servers, this is a "Self-Healing", autonomic computing algorithm.

- Complements known art, such as RR-DNS, which is used by most enterprise deployments to provide a one-to-many IP address mapping. As such, seamlessly coexists with enterprise architecture.

- This art can be done "under the hood" and doesn't require the expensive and time-consuming porting of operating-system present algorithms, drivers, and middleware across several vendors.

6. Identify the closest or most pertinent prior art that you are aware of.

None

7. Who is likely to want to use this invention or infringe the patent if one is obtained and how would infringement be detected?

The parties likely to infringe on this art are any company that typically would sell system solutions that would like to leverage the novel aspects of this invention. This could be enterprise computing companies like DELL, IBM, Compaq, HP, Sun, etc. One could easily determine if infringement is occurring by observing the mechanisms that are being used for interaction and configuration of the machines. Attachment of an In-Target Probe (ITP) would allow for examination of binary executable code and detection of infringement of this algorithm. The tracing of the algorithms used for resource allocation interfaces would easily determine violations.

**HAVE YOUR SUPERVISOR READ AND FORWARD IT ELECTRONICALLY
VIA E-MAIL TO "INVENTION DISCLOSURE SUBMISSION"**

DATE: _____ SUPERVISOR: _____

BY APPROVING, I (SUPERVISOR) ACKNOWLEDGE THAT I HAVE READ AND UNDERSTAND THIS
DISCLOSURE, AND RECOMMEND THAT THE HONORARIUM BE PAID